Weekly report (2013.9.23 ~9.29)

Done

1) Modify my paper for CAD Graphics 2013, re-format it according to IEEE Computer Society Proceedings Manuscript Formatting Guidelines and reduce it to 8 pages.

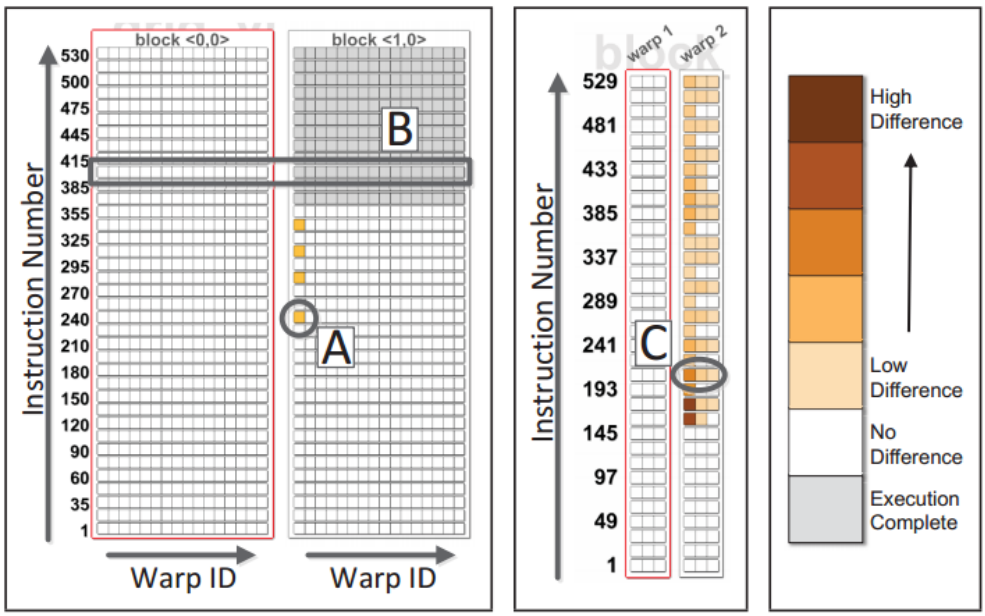
Figure 1 paper overview

2) Write a ".bat" for VisNgin Project, making all steps in "ReadMe.txt" an one-click operation. It simplifies the deploy of VisNgin, of which I think will be helpful for the new guys engaged in this project.

   Read the code of VisNgin, together with Tianye. It is almost clear except for the rendering part(OpenGL).

3) Read several relevant papers from vis conferences.

   [1]. "*A Visual Approach to Investigating Shared and Global Memory Behavior of CUDA Kernels*" (EuroVis 2013)



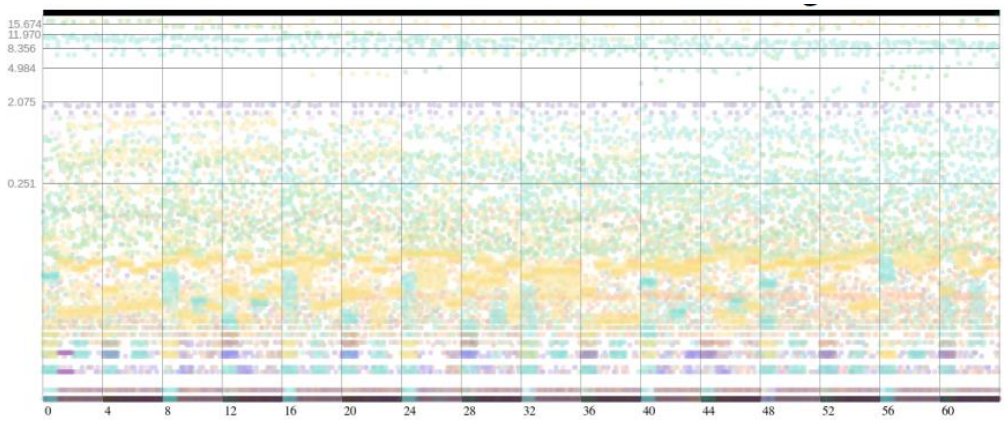
This paper proposed a method to visualize memory usage in CUDA applications. It first define metrics for comparing and find those representative ***warps***. Then, we can look into the selected warps and identify operations that are performance bottlenecks. It a

level-of-detail method, users can investigate from overview to details.

Its encoding and stories are both good, but it does not provide a complete analysis pipeline (i.e. how to choose the representative warps/blocks/instructions.)

I think it's a good topic that using visualization to help improve program efficiency, especially parallel programs. But with the premise that we know well about the specific parallel framework and exactly what to visualize, we can do some similar work when the time is ripe.

[2]. "*Visualizing Large-scale Parallel Communication Traces Using a Particle Animation Technique*" (EuroVis 2013)



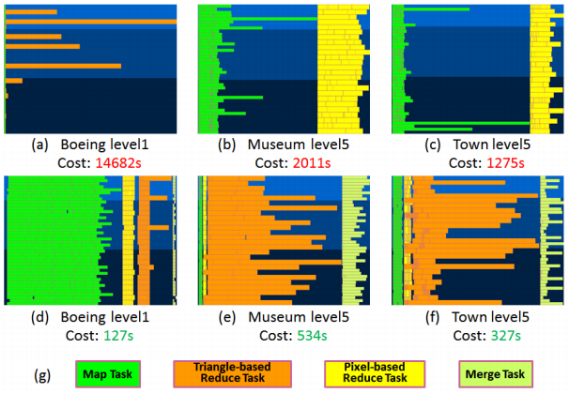
This paper visualizes the communication between processes in MPI applications. The main contribution is that such a particle animation based visualization can scale up to tens of thousands processors, while it can help discover problems in MPI programs.

The idea and technique is quite easy, but the authors have carefully designed the visualization at points such as transparency and overlap, thus making it clear and intuitive.

The result is just-so-so and the explain of those cases is somehow ambiguous in my opinion.

这两篇文章都属于 EuroVis 的 Performance 部分，共同点是都是借助可视化的手段去可视化一些影响并行程序运行的关键指标，从而帮助消除程序的瓶颈，改进程序的性能。这类可视化在基本可视化方法上没有什么创新，而是从实际问题出发，是目的驱动地去设计可视化，只要把问题有效地解决了，就是好的可视化。

我在之前的文章里也做了初级的尝试，当时也是本着想直观地看到并行程序执行情况的想法做的。而前面两篇文章，他们做得更底层（消息传递、内存使用），更能揭示出程序本质上的执行模式、运行瓶颈，因此也显得更具有说服力。



马老师的这篇文章前面部分直到可视化的设计，写得都很不错，很有道理。但是 case study 部分感觉比较弱，只是分析了看到的一些现象，多处使用 may/likely 这些词，并没有给出在这个可视化指导下优化的实际案例。感觉还没有到很实用的地步，包括另一篇文章也是。

之后如果有需要，觉得可以尝试下这方面的研究。

To Do

1) Try to find out the cause of the problem with the camera in VisNgin.
2) Continue implementing the demo of rendering a textured earth with Equalizer.